PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 11/20 | A1

(11) International Publication Number: WO 98/49620

(43) International Publication Date: 5 November 1998 (05.11.98)

(21) International Application Number: PCT/US98/08142

(22) International Filing Date: 22 April 1998 (22.04.98)

(30) Priority Data:
08/845,718 25 April 1997 (25.04.97) US

(71) Applicant: SYMBIOS, INC. [US/US]; 2001 Danfield Court, Fort Collins, CO 80525 (US).

(72) Inventor: DELANEY, William, P.; 14715 Siefkes, Wichita, KS 67230 (US).

(74) Agent: BAILEY, Wayne, P.; Symbios, Inc., 2001 Danfield Court, Fort Collins, CO 80525 (US).

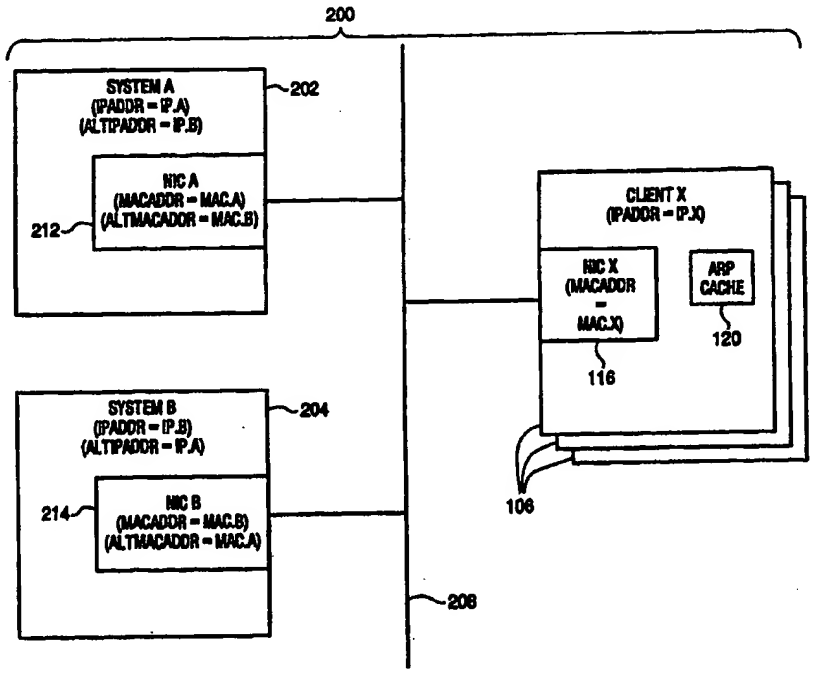
(81) Designated States: AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CU, CZ, DE, DK, EE, ES, FI, GB, GE, GH, GM, GW, HU, ID, IL, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, UA, UG, UZ, VN, YU, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SZ, UG, ZW), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: REDUNDANT SERVER FAILOVER IN NETWORKED ENVIRONMENT

(57) Abstract

Methods and associated apparatus for rapidly activating a standby server (204) to take over for a failed server (202) in a redundant server network environment. The methods of the present invention configure the NIC of the redundant (standby) server (204) to receive information having a MAC address corresponding to the main server in addition to its own unique MAC address. Multi-cast features common to commercially available NICs may be used for such configuration. The standby server (204) may normally process information addressed to its unique NIC MAC address and the associated logical address (e.g., IP address) while also monitoring (receiving) information addressed to the main server's NIC MAC address. When the standby server (204) takes over for the failed main server (202), it begins responding to packets received for the failed server's NIC MAC address. In the preferred embodiment of the invention, the standby server (204) reconfigures its NIC MAC addresses to receive packets addressed to the main server (202) only upon detection of a failure in the main server (202). Responses are generated by assuming the logical address identity of the failed server (202) but with the lower level (MAC) address uniquely assigned to the redundant server's NIC. Such replies cause clients (106) to immediately update any address cache (e.g., ARP tables) (102) without needing to await delays associated with error recovery timeouts. In addition, the present invention is applicable to a wider variety of network topologies, including ring topologies such as Token Ring and FDDI, as compared to prior techniques because the physical address generated by responses from the redundant server are consistent with its unique MAC address.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

REDUNDANT SERVER FAILOVER IN NETWORKED ENVIROMENT

1. Technical Field

This invention relates to distributed computing environments and in particular to methods and associated apparatus for activating a standby server upon takeover from a failed server in a redundant server environment with minimal delays imposed in client services and applicable to a wide variety of network topologies.

2. Background Art

Distributed computing environments, as the term is used herein, are those in which client processes request services from server processes. In particular, distributed computing environments are often used in conjunction with network communication media and protocols to enable distribution of the various communicating processes across physically remote nodes and locations. A server process may, for example be operable in a server computing node while a client process which requests services from the server process may be operable in the same node or in a remote node connected via communication networks.

Computing nodes in a distributed computing environment are connect to the network communication medium via network adapters or network interface cards (also referred to herein as NICs). A NIC provides circuits to receive and transmit data over the network communication medium on behalf of the computing node in which it is housed. As used herein, such a computing system may include general purpose computer systems (e.g., host systems) into which a NIC is inserted as well as peripheral devices with embedded NIC circuits which attach the peripheral to the network medium. A server or service process may be, for example, a file server providing coordinate access to files on behalf of a plurality of client processes, a print server providing printer functions to a plurality of client process, or any other function which provides services on behalf of a client process. A server

process may therefore be operable within a general purpose computing node or may be embedded within a special purpose server device. A client process is therefore any process which requests such services from a server process whether operable in a general purpose computing environment or embedded in a special purpose device.

Protocols used in network communication are often modeled as layers of modules. The lowest level layers manage the lowest levels of processing required to apply signals to the network communication medium destined to another node. Higher level modules manage correspondingly higher level functions relating to transferring information between two processes (e.g., routing of messages between a client node and a server node or error recovery retransmissions).

At the lowest layers of network communications in many common protocols used in network distributed computing environments (e.g., Ethernet, Token Ring, FDDI, etc.), the NIC has an address associated therewith often referred to as a media access address or MAC address (also referred to herein as a physical address). This address is programmed into the NIC typically at time of manufacture in accordance with industry standards which help assure a globally unique address is assigned each NIC. The MAC address is used to uniquely identify and distinguish the communicating nodes on the network medium. Blocks (packets) of information applied to the communication medium typically include address fields which identify the NIC which is the source of the packet and the NIC which is the destination of the packet. The MAC address of the NIC is therefore physical in nature in that it identifies the NIC globally regardless of the particular system in which it is operating.

Higher level protocols (e.g., TCP/IP) utilize other addresses (also referred to herein as logical addresses) within their portions of the data packet to identify higher level components of the network. In TCP/IP protocols, for example, a packet includes a source and destination IP address to identify the communicating nodes. These addresses are logical in that they may be

dynamically assigned to account for reconfiguration of the network topology and to logically group related nodes. Most protocols (e.g., TCP/IP) therefore include a mapping of the higher level protocol specific logical addresses (e.g., IP addresses) to the corresponding lower level physical addresses (e.g., NIC MAC addresses). A request is addressed to a server based upon its logical address (e.g., the IP address). This logical address is translated into a corresponding lower level (physical) MAC address in order to transmit the request to the proper computing node.

In many network protocols, portions of the protocol are defined for exchange of address mapping information so that changes in address mapping can be promulgated throughout the network. In TCP/IP protocol standards, for example, an address resolution protocol (ARP packets) is defined for use in transferring such address mapping information throughout the network. A first node sends out a packet on the network including a higher level logical address (e.g., an IP address in TCP/IP) for which it requires the low level physical address (e.g., NIC MAC address). Nodes in the network typically cache such information after acquiring it a first time to reduce the volume of such network traffic.

It is known in the art to provide redundancy as a means for improving reliability and availability of a computing application. In a client/server distributed environment, redundant server nodes are often utilized to help assure reliable access to the service(s) provided thereby. Typically, one server node or system provides a particular service while its redundant paired server node remains idle (with respect to provision of the same service). The idle second server node takes over the provision of the service when it senses that the first server has failed in some manner. The process of taking over service on behalf of a failed server is also referred to herein as "activating" the idle second server.

In general, it is a problem in such environments to efficiently permit one server node to take over for a redundant but failed server node with respect to network communications addressing. For example, in TCP/IP

communications, Veritas® Firstwatch® provides for redundant server take over in networked distributed computing environments by permitting the redundant server to assume the IP and MAC address identity of its associated failed server. However, as noted by Veritas®, this method is not usable in certain networked topologies such as Token Ring or FDDI rings. In such ring topologies, the physical position of a particular NIC relative to its neighbors on the ring is vital to operation of the ring. Reconfiguration of the ring topology may be required where, as in Firstwatch®, a NIC suddenly assumes a different NIC address identity due to redundant take over procedures.

In other prior techniques applied to TCP/IP networks, the redundant server simply assumes the IP address of the failed server when taking over from the failed server thereby creating a new association (mapping) between an IP address and a lower level MAC address. However, client processes attempting to use the services of the redundant server will continue to use the cached MAC address associated with the failed server's IP address. Information routed to the failed server at its IP address will therefore not be received by the redundant server because the MAC address used in the transmission is not the MAC address used by the redundant server's NIC card. Eventually, the client process node will timeout awaiting acknowledgment of information sent to the failed server's IP and MAC address. This timeout (often referred to in TCP/IP protocols as the ARP timeout) will cause the client to request an update of its cached IP/MAC address mapping. ARP messages are exchanged to update the clients ARP cache and processing then continues with the redundant server.

Such problems are not unique to TCP/IP protocol applications. More generally, it is a problem with redundant network servers to take over for one another in a manner which is rapid, transparent to client applications, and applicable to a wide variety of network applications and topologies.

It can be seen from the above discussion that a need exists for an improved method for activating a standby (redundant) server on a network when taking over for a failed server. Specifically, a need exists for a method

which permits takeover of a failed server by a redundant server regardless in a variety of network topologies and without need for awaiting error recovery timeout conditions.

3. Disclosure of Invention

The present invention solves the above and other problems, thereby advancing the state of the useful arts, by providing methods and associated apparatus for essentially immediate restoration of service to all clients in a wide variety of network topologies and applications. In particular, the methods of the present invention restore service to clients without need for the client processes to await error recovery timeout conditions and associated updates of their address mapping cache (e.g., ARP cache in TCP/IP protocols). The methods of the present invention are applicable in Ethernet as well as ring topologies (e.g., Token Ring and FDDI).

Specifically, methods of the present invention utilize multi-cast features of modern NICs to enable a NIC to receive information addressed to any of a plurality of MAC addresses. In particular, the NIC in a redundant server is configured to actively send and receive information using its own unique MAC address but is additionally configured to receive information addressed to the unique MAC address of the server for which it is the redundant mate. Multi-cast features are generally available on a number of commercially available NICs for enabling a NIC to "listen" to a plurality of MAC addresses. In true multi-cast applications, a multi-cast MAC address is used to broadcast information to a plurality of NICs all configured to receive information packets addressed to the multi-cast address as well as their own unique MAC address. Typically the multi-cast address is programmed into the NIC to provide flexible configuration of the multi-cast addressing.

The multi-cast address feature of the redundant server's NIC is programmed to receive packets addressed to the address of its paired server. When the redundant server determines that it must take over provision of the

service from its paired server, it assumes the IP address identity of the failed server and begins responding to packets addressed to the failed server's IP and MAC address. However, unlike prior techniques, responses to such received packets are transmitted using the failed server's IP address but the MAC address uniquely assigned to the redundant server's NIC. Replies received at the client node using the failed server's IP address but a new MAC address, namely the MAC address of the redundant server's NIC, may cause the requesting client to immediately update its address mapping cache (ARP cache) to reflect the new address mapping information. The client need not await error recovery timeout conditions before updating its ARP cache. This feature of the present invention helps alleviate delay problems inherent in prior solutions. Other clients, depending upon the system architecture of the client, may not update their ARP cache. Rather, they will continue to transmit messages to the physical address associated with the failed system. However, the redundant server will receive such messages and respond appropriately as above.

In addition, methods of the present invention are applicable to a wider variety of network topologies than are previous techniques. Specifically, the methods of the present invention are applicable to, for example, Ethernet, Token Ring, and FDDI network topologies. Whereas prior techniques are inapplicable to Token Ring and FDDI (as well as other ring topologies), the methods of the present invention utilize the preconfigured MAC address uniquely assigned to the redundant server's NIC when responding to requests directed to the failed server's IP address.

Preferably, the redundant servers operate in a symmetric manner in that each may serve as a standby by for the other. The definition of standby and primary server as used herein, is therefore a relative one which describe a role and mode of operation rather than necessarily a physical designation of the servers. Specifically, each server of a pair of redundant servers acts in the role of primary server with regard to services it provides to particular clients. Simultaneously, each server of the redundant pair of servers acts as

a standby server for the other. As a standby server it stands ready to provide services for the other server by assuming the identity of the primary server when the primary server has failed.

In a first embodiment of the present invention, both redundant servers continually monitor the physical address of the other so as to maintain a synchronized state in case a take over is required. The standby server in such an embodiment may therefore assume the identity of the failed server with complete knowledge of the present state of the failed server. Such an embodiment is preferred where the network protocols in use between the clients and servers are state based rather than state-less.

In the preferred embodiment where state-less protocols such as network file system protocols (NFS) are used, a standby server need not constantly monitor messages addressed to the primary server. Rather, the standby server takes over from the failed primary server when the primary server is determined to have failed. At the time of such take over processing the standby server assumes the identity of the primary server. This relieves the standby server from the overhead of receiving messages destined for the primary server during normal operation which are ignored (other than maintaining state synchronization with the primary server as above).

It is therefore an object of the present invention to provide methods and associated apparatus for activating a standby server in a redundant server network.

It is another object of the present invention to provide methods and associated apparatus for activating a standby server in a redundant server network having any of a number of topologies including Ethernet and ring topologies.

It is a further object of the present invention to provide methods and associated apparatus for activating a standby server in a redundant server network while minimizing delays in service to clients.

It is still another object of the present invention to provide methods and associated apparatus for activating a standby server by assuming the

physical address identity of a redundant primary server when the primary server fails.

It is still a further object of the present invention to provide methods and associated apparatus for activating a standby server by assuming the physical address identity of a redundant primary server when the primary server fails and for maintaining a synchronized state between the primary and standby servers for requests processed by the primary server.

It is yet a further object of the present invention to provide a method and associated apparatus for activating a standby server in a redundant server network while minimizing delays in service to clients due to take over of a failed server by the standby server.

The above and other objects, aspects, features and advantages of the present invention will become apparent from the following detailed description and the attached drawings.

4. Brief Description of the Drawings

FIG. 1 is a block diagram of a redundant server network environment operable in accordance with known techniques devoid of the methods and apparatus of the present invention;

FIG. 2 is a block diagram of a redundant server environment operable in accordance with the improved methods and apparatus of the present invention;

FIG. 3 is a flowchart describing the operation of a redundant server while operable in a standby mode and while operable to take over from a failed server for a state based protocol requiring synchronization between the servers; and

FIG. 4 is a flowchart describing the operation of a redundant server while operable in a standby mode and while operable to take over from a failed server for a state-less protocol requiring no synchronization between the servers.

5. Detailed Description of the Preferred Embodiments

While the invention is susceptible to various modifications and alternative forms, a specific embodiment thereof has been shown by way of example in the drawings and will herein be described in detail. It should be understood, however, that it is not intended to limit the invention to the particular form disclosed, but on the contrary, the invention is to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

FIG. 1 is a block diagram depicting a typical network distributed client/server distributed computing application 100. As noted above, client/server model computing environments are applicable to networked as well as non-networked computing environments. In general, client processes request and receive services by communicating with server processes. The various client and server processes may be operable within a single, stand-alone, non-networked computing environment or may be distributed over a plurality of networked computing nodes. Though the client/server distributed computing model covers such a broad range of computing environments, the problems addressed by the present invention stem from configurations in which redundant server processes are each operable on distinct computing nodes and a client process is operable on yet another computing node.

In FIG. 1, a service is provided by system A 102 (also referred to herein as a server or server node) to client X 106 via communications over network 108. System B 104 (also referred to herein as a standby server or a redundant server) stands ready to take over provision of the service from server 102 in response to a failure detected in server 102. All nodes (102, 104, and 106) are attached to the network 108 via network interface circuits 112, 114, and 116 (also referred to herein as NICs), respectively.

As noted above, it is common in many network protocols to associate a low level (physical) address with each device (NIC) attached to a network. NIC 112 is associated with a physical address (MAC address) of MAC.A, NIC

114 has a MAC address of MAC.B and NIC 116 has a MAC address of MAC.X. Further, as noted above, many network protocols (e.g., TCP/IP) associate a higher level (logical) address with computing nodes or devices in which a NIC is embedded or inserted. In TCP/IP protocols, for example, this logical address is referred to as an IP address. In FIG. 1, server 102 is associated with a logical address (e.g., IP address) of IP.A, server 104 is associated with a logical address of IP.B and client 106 is associated with a logical address of IP.X.

Service requests generated by client 106 include source and destination logical addresses indicating that the service request is from client 106 by its logical address IP.X and is destined to server 102 by its logical address IP.A. Network processing within client 106 eventually applies data representative of the service request to the communication medium of the network 108 via NIC 116. The data applied to the physical medium of network 108 includes the low level physical addresses of the source and destination of the information. MAC address MAC.X is identified as the physical address source of the information while MAC address MAC.A is identified as the physical address destination of the information on network 108. The request is received by server 102 via its NIC 112 and processed appropriately. A response is generated for transmission back to the requesting client 106 using the same logical and physical addresses supplied in the request (though the source and destination roles may be reversed as required by the particular protocols).

Server 104 is idle (with respect to the particular service nominally provided by server 102) awaiting an indication that server 102 has failed in some manner. Such indications are provided and sensed by well known means outside the scope of the present invention. For example, servers 102 and 104 may periodically communicate to determine each others respective operational status. Failure to appropriately respond to such an inquiry may suffice as an indication that the systems filing to respond has failed and that the redundant server should take over control of the service nominally

provided by the failed server. Those skilled in the art will recognize many equivalent methods for recognizing indications that a main server has failed so as to initiate a take over by a redundant or standby server.

Redundant server 104 takes over provision of the services nominally provided by server 102 when it senses a failure in the operation of server 102. In accordance with prior methods, redundant server 104 may adopt the identity of failed server 102 by assuming both the logical and physical address identity (IP.A and MAC.A) of failed server 102. As noted above, assuming the physical address identity of failed server 102 is not a viable option in certain common network topologies such as ring networks (e.g., Token Ring and FDDI). In such topologies, the physical address is tightly coupled to physical location of the device on the ring (i.e., the neighbors with which it is directly associated). Assumption of the physical address (MAC.A) in NIC 114 of server 104 may therefore cause problems in such ring topologies.

In accordance with other known techniques, redundant server 104 may adopt only the logical address of failed server 102 (e.g., IP.A). In so doing, it can process new service requests received through its NIC 114 having a logical address of failed server 102 (e.g., IP.A). However, client 106 is unaware of the take over by server 104 of failed server 102 and is therefore ignorant of the mapping of the logical address of server 102 (IP.A) to the physical address of NIC 114 in sever 104 (MAC.B).

Eventually client 106 will detect an error in transmission of a packet to the physical address of failed server 102. For example, a timeout condition is likely to occur after sufficient time has passed while client 106 waits for a response to its request. In response to such an error client 106 may invoke protocol messaging features to inquire as to a new mapping of the logical address of server 102 to a new physical address. In TCP/IP protocols such messaging is referred to as address resolution protocol or ARP. ARP messages are generated by client 106 in response to an error condition in communicating with server 102 and redundant server 104 responds with a

new physical address (MAC.B) to be associated with the logical address of failed server 102 (IP.A).

Client 106 includes ARP cache 120 wherein it store address mapping information received in its initial configuration and as modified over time by address resolution protocol message exchange over network 108. Client 106 relies on the information stored in ARP cache 120 to avoid excessive ARP message traffic being generated on network 108 thereby using available bandwidth. Information in ARP cache 120 is updated when new mapping information is made available to client 106 or at certain time based events (e.g., timeouts corresponding to error conditions in network message traffic). As noted elsewhere, these delays may be undesirable in certain high performance distributed applications. During these delay periods the service nominally provided by failed server 102 are simply unavailable to client 106.

Although redundant server 104 can assume the logical address identity of failed server 102, client 106 remains unaware of the address mapping change until the ARP cache 120 is updated. Therefore, redundant server 104 receive none of the requests from client 106 and hence can do nothing to process such requests.

These and other problems of prior redundant network server take over techniques are solved by the techniques of the present invention. FIG. 2 is a block diagram of a distributed computing networked application 200 similar in many respects to that of FIG. 1 but operable in accordance with the methods of the present invention. In particular, each server 202 and 204 is operable to take over operation of the other in case of failure. Each server 202 and 204 is configured with its own logical address, IP.A and IP.B, respectively. As in FIG. 1, each server 202 and 204 includes a NIC 212 and 214, respectively. Each NIC 212 and 214 is configured with its own unique physical address, MAC.A and MAC.B, respectively. These nominal addresses are used in normal operation to identify each server and its associated service in communication over network 208 with clients 106.

In addition to their respective nominal logical and physical addresses, each server 202 and 204 is configured to remember the logical address of the alternate server, namely IP.B and IP.A, respectively. In like manner, each NIC 212 and 214 is configured to additionally receive transmissions on network 208 destined for the alternate physical address, namely MAC.B and MAC.A, respectively. Each server 202 and 204 is therefore capable of receiving (e.g., snooping or monitoring) requests destined for the alternate server. Each server ignores requests received that are destined for the alternate server so long as the alternate server is operable to provide requisite services and a take over procedure is unnecessary. In other words, each server 202 and 204 receives information transmitted to either server but processes only those that are intended for its servicing under normal operating conditions.

When server 204 detects a failure in the operation of server 202 it takes over processing of requests destined to server 202. Server 204 assumes the logical address identity (e.g., IP.A) of the failed server 202. Since server 204 was monitoring requests destined to failed server 202 it may immediately process any outstanding request and return an appropriate response to requesting client 106.

The response from redundant server 204 to client 106 is generated with the assumed logical address identity of failed server 202 (e.g., IP.A) and the physical address of NIC 214 in redundant server 204 (e.g., MAC.B). As noted above, depending upon the internal architecture of client 106, client 106 may immediately recognize in the response packet that the mapping of a logical address for server 202 (e.g., IP.A) to a physical address has changed and should now be directed to the new physical address received as the source physical address in the response, namely MAC.B the physical address of redundant server 204. Further requests to logical address IP.A are therefore transferred via network 208 to server 204 rather than failed server 202. Other client architectures may not update their address mapping information based upon the response message. Rather, such clients will

continue to use the existing mapping information which continues to direct requests to the logical address and physical address of the failed server. However, redundant (standby) server 204 receives, processes, and responds to such messages as described above during such take over mode processing of requests on behalf of the failed server.

Client 106 receives its responses essentially without delay under this method of the present invention because redundant server 204 is prepared to respond to any monitored message destined to server 202 in response to detecting a failure in operation of server 202. This method of the present invention is usable in a wider variety of network topologies such as ring networks because transmissions from server 204 via its NIC 214 use the preconfigured nominal physical address associated therewith, namely MAC.B.

Redundant server 204 continues to process requests addressed to the logical address of failed server 202 until server 204 detects that server 202 is again operable and prepared to resume normal processing of service requests. When so sensed as operable, server 202 will begin processing requests directed to its original logical address (IP.A) in a manner analogous to the take over processing above. Specifically, server 202 will monitor messages sent to its logical address but to the physical address of server 204 (MAC.B). Server 202 will process the next such request (in cooperation with redundant server 204). The response generated by such processing will use the logical address of server 202 (IP.A) and its nominal physical address (MAC.A). As above, client 106 will immediately detect the change in logical to physical address mapping reflected in the response received and update its ARP cache 120 accordingly.

Those skilled in the art will recognize that the operation of servers 202 and 204 is essentially symmetric. Each acts as a redundant service provider for the other. Each performs a take over process when it senses the other has failed. Each permits the other to resume processing of requests by

reverting the address mapping when the other is sensed to be again operational.

All address changes required for the take over and reversion processing between the two servers are performed as described above such that all information transmitted from a server uses the nominal physical address for that server. This feature allows the method of the present invention to be applied to ring topology networks such as Token Ring and FDDI in addition to Ethernet bus topologies. In addition, client 106 is notified immediately of the address mapping changes required in its ARP cache 120 by the first response returned from the redundant controller 204 following its take over from server 202. This feature eliminates delays incurred by previous techniques wherein the client would be unaware of the address change until potentially lengthy timeout delays when the client would naturally seek to update ARP cache information (e.g., in response to transmission and retry timeout errors).

FIG. 3 is a flowchart describing the methods of the present invention operable within server 202 and 204 to perform the take over processing as described with respect to FIG. 2 above. As noted above, the processing within each of a pair of redundant servers is (typically) symmetric. Each server provides its own particular services and serves as a redundant server to take over provision of services by the other in case of failure. The processing described in FIG. 3 is therefore applicable to either server. In particular, the method is preferably operable simultaneously within both servers 202 and 204 of FIG. 2 wherein each provides its own services to requesting clients and also provides a standby (redundant) service for the services of the other in case of failure. Those skilled in the art will recognize that the methods described herein are equally applicable if only one of the pair of servers 202 and 204 performs the role of redundant server. Similarly, the methods described herein may be trivially extended by those skilled in the art to encompass more than two redundant servers. Any number of redundant servers may be operable in accordance with the methods of the

present invention by appropriately coordinating among redundant server which will assume the identity of the failed server. FIG. 3 describes the processing of the methods of the present invention with respect to a pair of server for simplicity of presentation. Specifically, FIG. 3 describes the operation of the methods of the present invention from the perspective of system A (server 202 of FIG.2) wherein it acts as a redundant server for system B (server 204 of FIG. 2).

FIG. 3 describes a method of the present invention most applicable to state based network protocols in which messages are potentially interdependent upon one another. Exemplary of such a state based protocol is the Microsoft® Windows® networking protocol used for file and printer sharing among personal computers. Under this protocol a newly received message may require knowledge of a state determined by processing of preceding messages. In such protocol applications, the standby server needs to monitor message traffic destined to the primary server even during normal processing by the primary server. Messages received by the standby server but destined for the operational primary server are processed to the extent that any requisite state change information pertaining to the protocol is derived. No replies are generated by the standby server by virtue of such processing nor does the standby server generate any other side effects from the processing of the request. Only the state change information is gathered so as to maintain synchronization between the standby and primary servers. The method of FIG. 3 is required where state based protocols are applied to the network but is also operable in application where state-less protocols are utilized.

Element 300 is first operable to determine the physical address (MAC.B) for the alternate system B. This may be determined by any form of communication between system A and system B. For example, the two servers may exchange configuration information at initialization via the common network 208. Alternatively, there may be a dedicated communication channel between systems A and B for purposes of such

configuration communications. Element 302 then logically binds or associates the logical address (IP.A) for system A to the physical address for system A's NIC (MAC.A). This binding is a logical configuration which establishes the logical to physical address mapping for information to received or transmitted by system A via network 208.

Element 304 then provides configuration information to system A's NIC to enable the NIC to receive information on either of the two physical addresses associated with the redundant pair of servers (MAC.A or MAC.B). As noted, standard multi-cast features of most commercially available NICs may be used to receive information on either of two physical addresses. One exemplary such NIC device utilized in the preferred embodiment of the present invention is the ***Digital Semiconductor 21140A PCI Fast Ethernet LAN Controller*** made by Digital Equipment Corporation (DEC) of Maynard Massachusetts. This integrated circuit provides high speed Ethernet communications including programmable multi-cast features to enable operation of the methods of the present invention.

The method of the present invention as shown in FIG. 3 then enters a normal processing mode awaiting failure of system B which, in turn, initiates a take over process. Specifically, element 306 is operable to determine if such failure indicia from system B has been sensed. As noted above, any of several well known methods for detecting a failure may be utilized in conjunction with the methods of the present invention. For example, so called watchdog timer messages may be exchanged between the redundant servers on a frequent periodic basis. Failure to receive such a watchdog message from the alternate system is an indication that a take over process is required. Such messages may be exchanged via the common network 208 of FIG. 2 or over a dedicated communication channel among the redundant controllers.

So long as element 306 does not sense such a failure indication, elements 308 and 310 are operable to process normal requests received by system A through its NIC via network 208 addressed to its nominal logical and physical address (e.g., IP.A and MAC.A). Specifically, element 308

processes normal requests destined for system A by performing the requested processing and returning normal responses resulting therefrom. Element 310 assures that information received via system A's NIC destined for system B's physical address (MAC.B) is processed to the extent necessary to synchronize with the state processing of the network protocol in use. Beyond such synchronization, the message is otherwise ignored in the sense that no reply is generated or transmitted to the requesting client by the standby server. Depending on the computing environment (e.g., operating system features and networking features), element 310 may represent proactive steps to assure such packets are ignored or discarded within system A. In other environments, such messages may be discarded at lower levels of the computing environment (e.g., at the NIC or within various of the layers of the network processing modules - the protocol stack). Essentially, element 310 is shown to stress that packets received by system A which are destined for system B are processed differently than packets destined to system A during normal processing so that system B may process such requests in its normal course of processing. Normal processing continues iterativley with elements 306-310 until element 306 senses an indication of failure in the alternate system B.

Responsive to sensing of a failure by element 306, elements 312-316 are iteratively operable to perform redundant mode processing within system thereby processing requests destined to the failed system B as well as those destined to system A for normal processing. Specifically, element 312 is operable in like manner to element 308 to process any requests received by system A's NIC which are addressed to system A for normal processing. Such requests are processed by performing the requisite processing and returning any required messages to the requesting client. Such requests and responses use the nominal logical and physical addresses of system (IP.A and MAC.A).

Element 312 is also operable to process requests received on system A's physical address (MAC.A) but directed to system B's logical address

(IP.B). As noted below, clients adjust their respective addressing mapping tables in response to particular responses generated by element 314. Requests generated by clients following such adjustments to their address mapping tables are directed to system B's logical address (IP.B) but systems A's physical address (MAC.A). Such requests are processed on behalf of the failed system B as though they were normally directed for processing to system A. As noted elsewhere, some client may not adjust their respective address mapping tables. Such clients will continue to address requests to the address of failed system B (IP.B and MAC.B). None the less, element 314 will process the requests and continue to return required responses to the requesting clients.

Element 314 is operable within system A to process requests destined to system B and normally ignored by element 310 during normal processing. During redundant processing, element 314 receives and processes requests addressed to the logical address of failed system B (IP.B). Initially such requests are addressed to the logical and physical address of system B (IP.B and MAC.B). However, in system A generates and returns any required response to the request using the logical address of system B but the physical address nominally assigned to system A (i.e., its own physical address MAC.A). As noted above, such a response which associates systems A's physical address (MAC.A) with failed system B's logical address (IP.B) will cause the requesting client to immediately adjust its address mapping table (ARP cache) to account for the revised mapping. Other clients which request services from failed system B will also immediately adjust their respective address mapping tables as they receive their first response from system A indicative of the addressing change. Eventually all clients will have adjusted their address mapping tables in accordance with the responses generated by element 314 of FIG. 3. Further requests from such clients to system B (at logical address IP.B) will utilize system A's physical address (MAC.A) in accordance with their adjusted address mapping tables. System A will process such requests

Element 316 is then operable to detect whether it is now possible to return to normal processing mode. Specifically, element 316 determines whether indicia of a return to normal processing by failed system B has been sensed. As noted above, watchdog timer messaging may provide both an indication of a failure as well as an indication of a return to operational status in a previously failed system. Those skilled in the art will recognize several equivalent methods for sensing indicia of restoration of a failed system to an operational state. If element 316 does not sense such an indication, elements 312-316 are iteratively operable to continue processing requests directed to either system A or failed system B.

When element 316 determines that system B has been restored to an operation status, processing returns to element 306 to continue processing requests in normal mode wherein requests directed to system B are processed by system B. To achieve the switch back to normal processing within restored system B, system B must be prepared to receive a request addressed to its logical address (IP.B) but still addressed to the physical address of system A (MAC.A) due to the address mapping changes induced by system A's response to a clients request. Restoration of normal processing is therefore analogous to the processing shown in FIG. 3 to perform an initial take over of operations for a failed system. Specifically, system B will receive requests addressed to its logical address (IP.B) with a physical address of either system A or B (MAC.A or MAC.B). Responses generated to processing of such requests will utilize the nominal logical and physical addresses of system B (IP.B and MAC.B) to thereby force another immediate update of the requesting client's address mapping tables.

FIG. 4 is a flowchart describing a preferred embodiment of the methods of the present invention operable within server 202 and 204 to perform the take over processing as described with respect to FIG. 2 above. As noted above, the processing within each of a pair of redundant servers is (typically) symmetric. Each server provides its own particular services and serves as a redundant server to take over provision of services by the other in

case of failure. The processing described in FIG. 4 is therefore applicable to either server. In particular, the method is preferably operable simultaneously within both servers 202 and 204 of FIG. 2 wherein each provides its own services to requesting clients and also provides a standby (redundant) service for the services of the other in case of failure. Those skilled in the art will recognize that the methods described herein are equally applicable if only one of the pair of servers 202 and 204 performs the role of redundant server. Similarly, the methods described herein may be trivially extended by those skilled in the art to encompass more than two redundant servers. Any number of redundant servers may be operable in accordance with the methods of the present invention by appropriately coordinating among redundant server which will assume the identity of the failed server. FIG. 4 describes the processing of the methods of the present invention with respect to a pair of server for simplicity of presentation. Specifically, FIG. 4 describes the operation of the methods of the present invention from the perspective of system A (server 202 of FIG.2) wherein it acts as a redundant server (standby server) for system B (server 204 of FIG. 2).

FIG. 4 describes a preferred embodiment of a method of the present invention wherein exclusively state-less protocols are used in communicating over the network. Exemplary of such a state based protocol is the network file system (NFS) networking protocol used for file sharing among computers. Under this protocol a newly received message requires no state information derived from processing of preceding messages. In such state-less protocol applications, the standby server need not monitor message traffic destined to the primary server during normal processing by the primary server. Specifically, in the preferred method of FIG. 4, each server receives packets addressed only to its nominal physical and logical address during normal operation. Only when a failure is sensed in the primary server (system B as described in FIG. 4) does the standby server (system A) reconfigure its NIC to receive packets destined to either physical address (MAC.A or MAC.B).

Element 402 is first operable to logically bind or associate the logical address (IP.A) for system A to the physical address for system A's NIC (MAC.A). This binding is a logical configuration which establishes the logical to physical address mapping for information to received or transmitted by system A via network 208. Element 404 then provides configuration information to system A's NIC to enable the NIC to receive information on its nominally assigned physical address (MAC.A).

The method of the present invention as shown in FIG. 4 then enters a normal processing mode awaiting a failure of system B which, in turn, initiates a take over process. Specifically, element 406 is operable to determine if such failure indicia from system B has been sensed. As noted above, any of several well known methods for detecting a failure may be utilized in conjunction with the methods of the present invention. For example, so called watchdog timer messages (also referred to as heartbeat messages) may be exchanged between the redundant servers on a frequent periodic basis. Failure to receive such a watchdog message from the alternate system is an indication that a take over process is required. Such messages may be exchanged via the common network 208 of FIG. 2 or over a dedicated communication channel among the redundant controllers.

So long as element 406 does not sense such a failure indication, element 408 is operable to process normal requests received by system A through its NIC via network 208 addressed to its nominal logical and physical address (e.g., IP.A and MAC.A). Specifically, element 408 processes normal requests destined for system A by performing the requested processing and returning normal responses resulting therefrom. Processing then continues iteratively performing elements 406 and 408 until a failure is sensed in system B.

Responsive to sensing of a failure by element 406, elements 410-416 are iteratively operable to perform redundant mode processing within system thereby processing requests destined to the failed system B as well as those destined to system A for normal processing. Specifically, element 410 and

412 determine the present physical address (MACADDR) assigned to system B (MAC.B) and reconfigure system A's NIC to receive messages address to either physical address (MAC.A or MAC.B). Element 412 is then operable in like manner to element 408 to process any requests received by system A's NIC which are addressed to either system A for normal processing or addressed to system B for redundant (take over) processing. Such requests are processed by performing the requisite processing and returning any required messages to the requesting client. Such requests and responses use the logical address supplied by the request (IP.A if directed to system A or IP.B if directed to system B) and the physical address of system A (MAC.A). As noted below, clients may adjust their respective addressing mapping tables in response to particular responses generated by element 414. Requests generated by clients following such adjustments to their address mapping tables are directed to system B's logical address (IP.B) but systems A's physical address (MAC.A). Such requests are processed on behalf of the failed system B as though they were normally directed for processing to system A. Where clients do not adjust their address mapping information, requests will continue to be directed to the logical and physical address of the failed primary system B. Such requests will non the less be processed by the redundant (standby) processing of element 414.

Element 416 is then operable to detect whether it is now possible to return to normal processing mode. Specifically, element 416 determines whether indicia of a return to normal processing by failed system B has been sensed. As noted above, watchdog timer messaging may provide both an indication of a failure as well as an indication of a return to operational status in a previously failed system. Those skilled in the art will recognize several equivalent methods for sensing indicia of restoration of a failed system to an operational state. If element 416 does not sense such an indication, received message continue to be processed by iterative performance of element 414.

When element 416 determines that system B has been restored to an operation status, processing returns to element 404 to continue processing

requests in normal mode wherein requests directed to system B are processed by system B. In particular, element 404 reconfigures system A's NIC to return to normal operation wherein only requests directed to system A (IP.A and MAC.A) are processed. Elements 406 and 408 then resume normal mode processing until another failure is detected.

As noted above, a key feature of both methods described above with respect to FIGS. 3 and 4 is that replies returned from the standby server when performing requests on behalf of the failed primary server are returned with a physical address corresponding to the nominal physical address of the standby server. This key feature enables the methods of the present invention to be utilized with a wider range of network topologies and media than was possible with prior approaches.

While the invention is susceptible to various modifications and alternative forms, a specific embodiment thereof has been shown by way of example in the drawing and has been described in detail. It should be understood, however. that it is not intended to limit the invention to the particular form disclosed, but on the contrary, the invention is intended to cover all modifications, equivalents, and alternatives falling within the spirit and scope of the invention as defined by the appended claims.

CLAIMS

What is claimed is:

1. In a networked client/server computing environment having a client and a plurality of servers interconnected via a network communication medium, a method for controlling operation of a standby server of said plurality of servers in response to status changes of a primary server of said plurality of servers, said method comprising the steps of:

configuring said standby server to receive information from said network having a first network physical address;

detecting, within said standby server, a failure state of said primary server;

reconfiguring said standby server, in response to detecting said failure state, to receive information from said network having said first network physical address or having a second network physical address corresponding to said primary server;

receiving, within said standby server, client requests which are addressed to said second network physical address;

processing said client requests within said standby server; and

transmitting via said network replies to said client requests in response to said processing, wherein said replies are indicative of said first network physical address.

2. The method of claim 1 wherein said standby server includes a network interface circuit for connecting said standby server to said network and wherein the step of assigning said first network physical address comprises the step of:

configuring said network interface circuit with a predetermined value of said first network physical address.

3. The method of claim 2 wherein the step of assigning said second network physical address comprises the step of:

programming a multi-cast address in said network interface circuit to receive information transmitted via said network to said second network physical address.

4. The method of claim 1 wherein information transmitted over said network includes a destination logical address value and wherein the method further comprises the steps of:

assigning a first network logical address to said primary server; and

assigning a second network logical address to said standby server.

5. The method of claim 4 wherein said received client requests have a destination logical address equal to said first network logical address.

6. The method of claim 5 wherein said replies have a destination logical address equal to said first network logical address.

7. The method of claim 1 further comprising the steps of:

detecting, within said standby server, an operational state of said primary server following detection of said failure state; and

reconfiguring said standby server, in response to detecting said failure state, to receive information from said network having said first network physical address and to ignore information having said second network physical address.

8. In a networked client/server computing environment having a client and a plurality of servers interconnected via a network communication medium, a method for controlling operation of a standby server of said plurality of servers in response to status changes of a primary server of said plurality of servers, said method comprising the steps of:

configuring said standby server to receive information from said network having a first network physical address and to receive information from said network having a second network physical address corresponding to said primary server;

receiving, within said standby server, client requests which are addressed to said second network physical address;

detecting, within said standby server, a failure state of said primary server;

processing said client requests within said standby server in response to detecting said failure state; and

transmitting via said network replies to said client requests in response to said processing, wherein said replies are indicative of said first network physical address.

9. The method of claim 8 wherein said standby server includes a network interface circuit for connecting said standby server to said network and wherein the step of configuring said standby server comprises the steps of:

configuring said network interface circuit with a predetermined value of said first network physical address; and

programming a multi-cast address in said network interface circuit to receive information transmitted via said network to said second network physical address.

10. The method of claim 8 wherein information transmitted over said network includes a destination logical address value and wherein the method further comprises the steps of:

assigning a first network logical address to said primary server; and

assigning a second network logical address to said standby server.

11. The method of claim 10 wherein said received client requests have a destination logical address equal to said first network logical address.

12. The method of claim 11 wherein said replies have a destination logical address equal to said first network logical address.

13. The method of claim 8 further comprising the step of:

ignoring said client requests within said standby server in response to sensing that said primary server is in an operational status.

14. A computer readable storage medium tangibly embodying programmed instructions for performing a method for controlling operation of a standby server in a computing environment having a client and a plurality of servers interconnected via a network communication medium in response to status changes of a primary server of said plurality of servers, the method comprising the steps of:

configuring said standby server to receive information from said network having a first network physical address;

detecting, within said standby server, a failure state of said primary server;

reconfiguring said standby server, in response to detecting said failure state, to receive information from said network having said first network physical address or having a second network physical address corresponding to said primary server;

receiving, within said standby server, client requests which are addressed to said second network physical address;

processing said client requests within said standby server; and

transmitting via said network replies to said client requests in response to said processing, wherein said replies are indicative of said first network physical address.

15. The storage medium of claim 14 wherein said standby server includes a network interface circuit for connecting said standby server to said network

and wherein the method step of assigning said first network physical address comprises the step of:

configuring said network interface circuit with a predetermined value of said first network physical address.

16. The storage medium of claim 15 wherein the method step of assigning said second network physical address comprises the step of:

programming a multi-cast address in said network interface circuit to receive information transmitted via said network to said second network physical address.

17. The storage medium of claim 14 wherein information transmitted over said network includes a destination logical address value and wherein the method further comprises the steps of:

assigning a first network logical address to said primary server; and

assigning a second network logical address to said standby server.

18. The storage medium of claim 17 wherein said received client requests have a destination logical address equal to said first network logical address.

19. The storage medium of claim 18 wherein said replies have a destination logical address equal to said first network logical address.

20. The storage medium of claim 14 wherein the method further comprises the steps of:

detecting, within said standby server, an operational state of said primary server following detection of said failure state; and

reconfiguring said standby server, in response to detecting said failure state, to receive information from said network having said first network physical address and to ignore information having said second network physical address.

21. A computer readable storage medium tangibly embodying programmed instructions for performing a method for controlling operation of a standby server in a computing environment having a client and a plurality of servers interconnected via a network communication medium in response to status changes of a primary server of said plurality of servers, the method comprising the steps of:

configuring said standby server to receive information from said network having a first network physical address and to receive information from said network having a second network physical address corresponding to said primary server;

receiving, within said standby server, client requests which are addressed to said second network physical address;

detecting, within said standby server, a failure state of said primary server;

processing said client requests within said standby server in response to detecting said failure state; and

transmitting via said network replies to said client requests in response to said processing, wherein said replies are indicative of said first network physical address.

22. The storage medium of claim 21 wherein said standby server includes a network interface circuit for connecting said standby server to said network and wherein the method step of configuring said standby server comprises the steps of:

configuring said network interface circuit with a predetermined value of said first network physical address; and

programming a multi-cast address in said network interface circuit to receive information transmitted via said network to said second network physical address.

23. The storage medium of claim 21 wherein information transmitted over said network includes a destination logical address value and wherein the method further comprises the steps of:

assigning a first network logical address to said primary server; and

assigning a second network logical address to said standby server.

24. The storage medium of claim 23 wherein said received client requests have a destination logical address equal to said first network logical address.

25. The storage medium of claim 24 wherein said replies have a destination logical address equal to said first network logical address.

26. The storage medium of claim 21 wherein the method further comprises the step of:

ignoring said client requests within said standby server in response to sensing that said primary server is in an operational status.

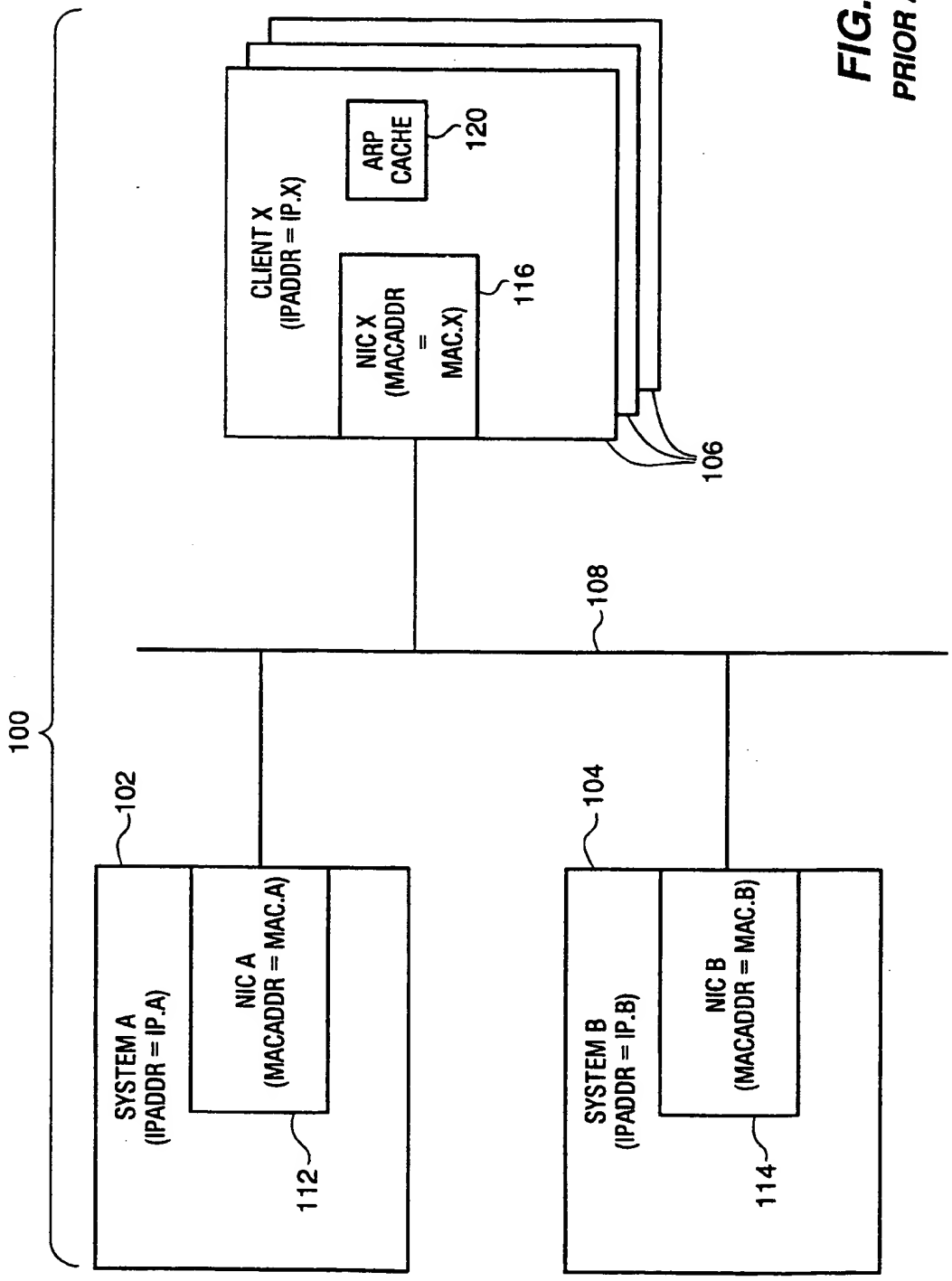


FIG. 1
PRIOR ART

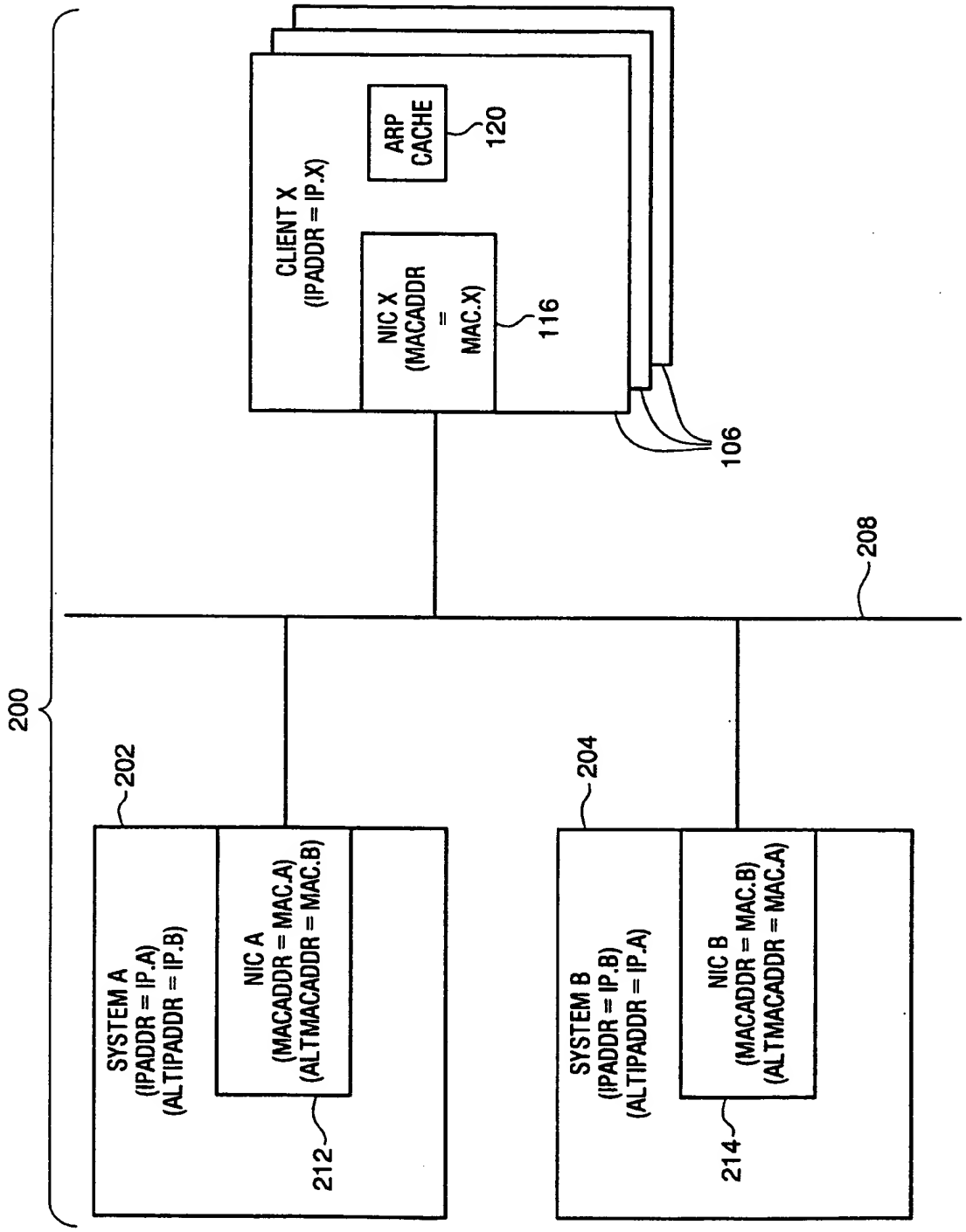

200
202
SYSTEM A
(IPADDR = IP.A)
(ALTIPADDR = IP.B)
NIC A
(MACADDR = MAC.A)
(ALTMACADDR = MAC.B)
212
204
SYSTEM B
(IPADDR = IP.B)
(ALTIPADDR = IP.A)
NIC B
(MACADDR = MAC.B)
(ALTMACADDR = MAC.A)
214
208
CLIENT X
(IPADDR = IP.X)
ARP CACHE
120
NIC X
(MACADDR = MAC.X)
116
106

FIG. 2

FIG. 3

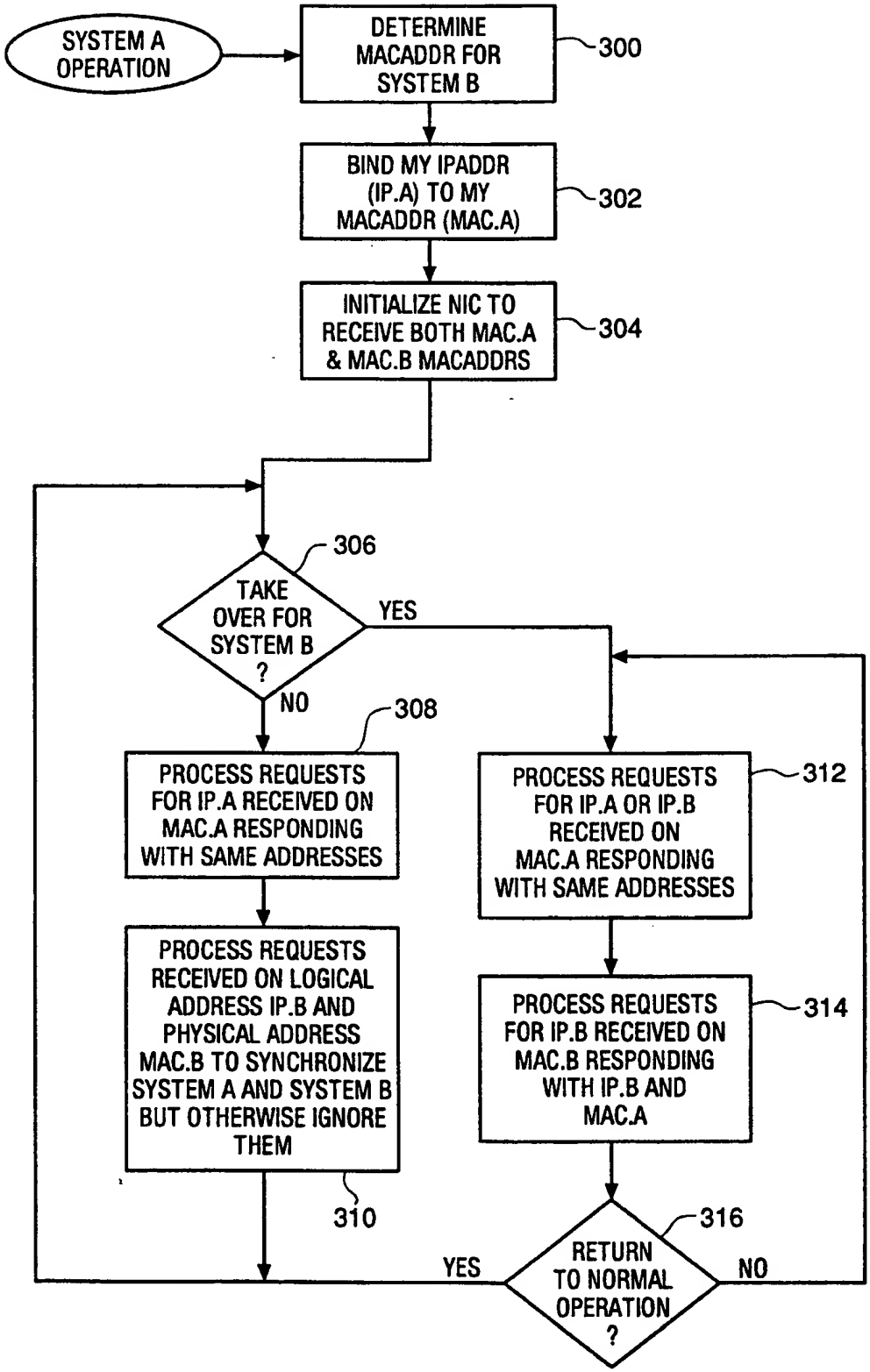
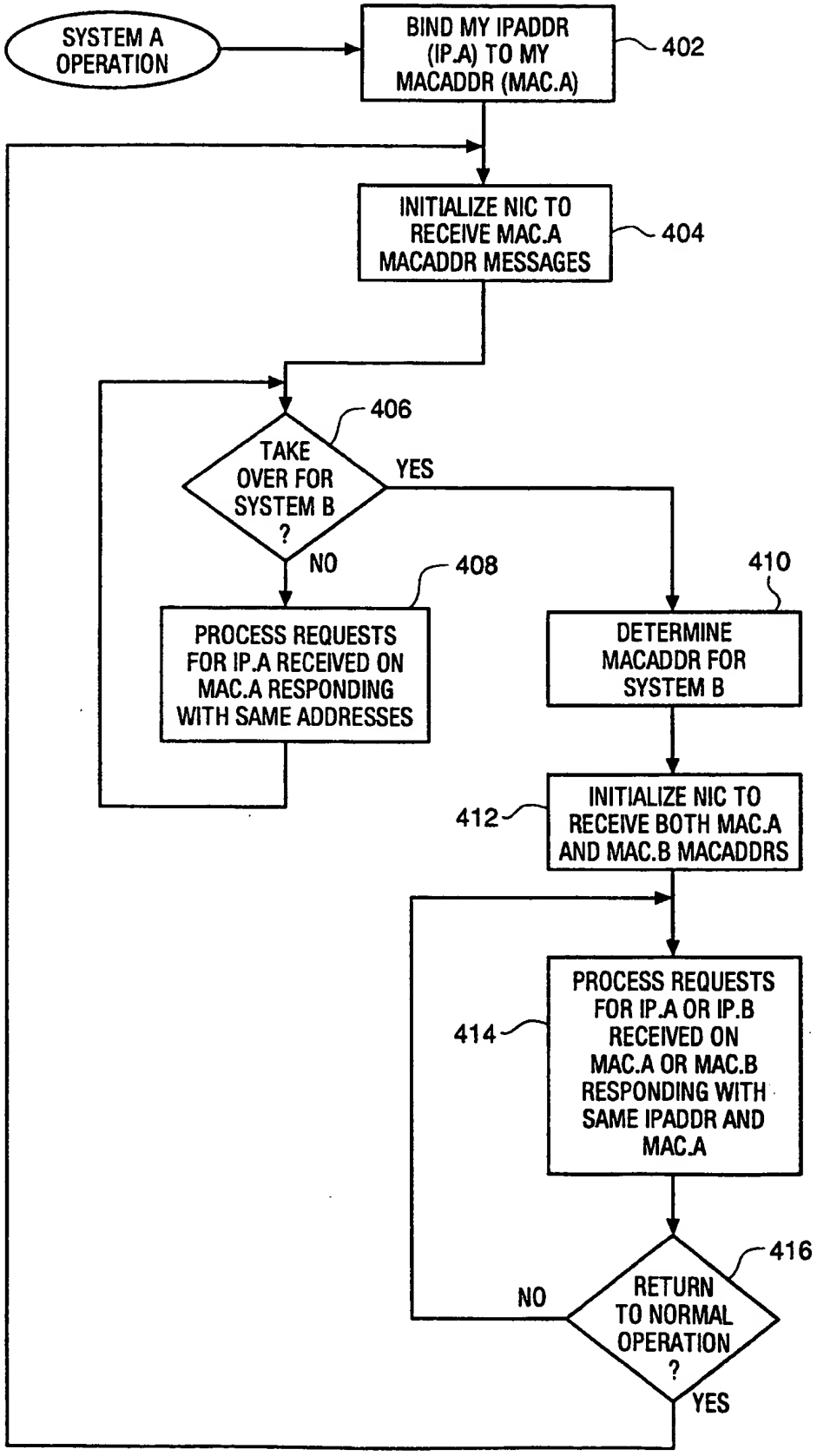

SYSTEM A OPERATION
DETERMINE MACADDR FOR SYSTEM B
300
BIND MY IPADDR (IP.A) TO MY MACADDR (MAC.A)
302
INITIALIZE NIC TO RECEIVE BOTH MAC.A & MAC.B MACADDRS
304
306
TAKE OVER FOR SYSTEM B ?
YES
NO
308
PROCESS REQUESTS FOR IP.A RECEIVED ON MAC.A RESPONDING WITH SAME ADDRESSES
PROCESS REQUESTS RECEIVED ON LOGICAL ADDRESS IP.B AND PHYSICAL ADDRESS MAC.B TO SYNCHRONIZE SYSTEM A AND SYSTEM B BUT OTHERWISE IGNORE THEM
310
312
PROCESS REQUESTS FOR IP.A OR IP.B RECEIVED ON MAC.A RESPONDING WITH SAME ADDRESSES
314
PROCESS REQUESTS FOR IP.B RECEIVED ON MAC.B RESPONDING WITH IP.B AND MAC.A
316
RETURN TO NORMAL OPERATION ?
YES
NO

FIG. 4


SYSTEM A OPERATION
BIND MY IPADDR (IP.A) TO MY MACADDR (MAC.A)
402
INITIALIZE NIC TO RECEIVE MAC.A MACADDR MESSAGES
404
406
TAKE OVER FOR SYSTEM B ?
YES
NO
408
PROCESS REQUESTS FOR IP.A RECEIVED ON MAC.A RESPONDING WITH SAME ADDRESSES
410
DETERMINE MACADDR FOR SYSTEM B
412
INITIALIZE NIC TO RECEIVE BOTH MAC.A AND MAC.B MACADDRS
414
PROCESS REQUESTS FOR IP.A OR IP.B RECEIVED ON MAC.A OR MAC.B RESPONDING WITH SAME IPADDR AND MAC.A
416
RETURN TO NORMAL OPERATION ?
NO
YES

INTERNATIONAL SEARCH REPORT

Ir. ıtional Application No
PCT/US 98/08142

A. CLASSIFICATION OF SUBJECT MATTER
IPC 6 G06F11/20

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
IPC 6 G06F H04L

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category °	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	US 5 513 314 A (KANDASAMY ET AL.) 30 April 1996 see the whole document	1-26
A	WO 92 18931 A (EASTMAN KODAK COMPANY) 29 October 1992 see the whole document	1-26
A	US 5 590 285 A (KRAUSE ET AL.) 31 December 1996	

☐ Further documents are listed in the continuation of box C.

☒ Patent family members are listed in annex.

° Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
11 August 1998	19/08/1998

Name and mailing address of the ISA	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Tx. 31 651 epo nl, Fax: (+31-70) 340-3016	Absalom, R

1

Form PCT/ISA/210 (second sheet) (July 1992)

INTERNATIONAL SEARCH REPORT

Information on patent family members

In. .tional Application No
PCT/US 98/08142

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 5513314 A	30-04-1996	AU 4767796 A	14-08-1996
		CA 2211654 A	01-08-1996
		EP 0806010 A	12-11-1997
		WO 9623259 A	01-08-1996
WO 9218931 A	29-10-1992	EP 0536375 A	14-04-1993
		JP 5508506 T	25-11-1993
US 5590285 A	31-12-1996	AU 680931 B	14-08-1997
		AU 7404394 A	28-02-1995
		EP 0664906 A	02-08-1995
		EP 0713309 A	22-05-1996
		JP 8502613 T	19-03-1996
		WO 9504322 A	09-02-1995
		US 5535338 A	09-07-1996